REMARKS/ARGUMENTS

Claims 1-5, 7 and 20-31 are pending in the present application. Claim 1 was amended and claims 20-31 were added. No new matter has been added.

I. 35 U.S.C. § 103, Obviousness

The Examiner has rejected claims 1-5 and 7 under 35 U.S.C. § 103 as being unpatentable over Sprecher (art of record, U.S. Patent No. 6,948,059) in view of U.S. Patent No. 7,069,541 to Dougherty et al. (art made of record, hereafter "Dougherty"). This rejection is respectfully traversed.

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). For an invention to be prima facie obvious, the prior art must teach or suggest all the claim limitations. *In re Royka*, 490 F.2d 981, 180 U.S.P.Q. 580 (CCPA 1974). The Examiner has failed to meet this burden in the present case. More specifically, the Examiner has failed to teach or suggest all the claim limitations, and the Examiner has failed to show some motivation or incentive to one of ordinary skill in the art to modify or combine the references.

In *Graham* v. *John Deere Co. of Kansas City*, 383 U. S. 1 (1966), the Supreme Court set out a framework for applying the statutory language of §103. The Court stated:

> Under §103, the scope and content of the prior art are to be determined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background the obviousness or nonobviousness of the subject matter is determined. Such secondary considerations as commercial success, long felt but unsolved needs, failure of others, etc., might be utilized to give light to the circumstances surrounding the origin of the subject matter sought to be patented.

Id., at 1718.

In other words, these factual inquires include (1) determining <u>the scope and content</u> of a patent claim and the prior art relative to a claim in the application at issue; (2) determining the differences between the <u>scope and content</u> of the patent claim and the prior art as determined in

(1); (3) determining the level of ordinary skill in the pertinent art; and (4) evaluating any objective indicia of non-obviousness.

A. The Prior Art Must Teach Each and Every Element of the Claimed Invention

Amended independent claim 1 recites the following:

A method for testing a compatibility of software modules, the method comprising computer implemented steps of:

responsive to receiving a request to install a new software module in a data processing system, performing an inventory on an existing set of software modules resident in the data processing system;

referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with each software module in the existing set of software modules;

responsive to a determination that the new software module is not known to function compatibly with each software module in the existing set of software modules, determining whether to test the new software module in a test data processing system in combination with the existing set of software modules;

responsive to a determination to test the new software module in the test data processing system in combination with the existing set of software modules, identifying an operating environment of the data processing system;

installing the new software module, the identified environment, and the existing set of software modules on the test data processing system;

testing the new software module in combination with the existing set of software modules on the test data processing system;

responsive to a test result indicating that the new software module is compatible with the existing set of software modules, adding a new combination indicating the compatibility to the knowledge base; and

installing the new software module in the data processing system.

1. "responsive to a determination to test the new software module in the test data processing system in combination with the existing set of software modules, identifying an operating environment of the data processing system"

An element of claim 1 recites "responsive to a determination to test the new software module in the test data processing system in combination with the existing set of software modules, identifying an operating environment of the data processing system[.]"

The Examiner states that *Dougherty* at Figure 1 and column 7, line 52 through column 8, line 28 teaches this element of claim 1. *Dougherty* at column 7, line 52 through column 8, line 28 states the following:

Referring back to FIG. 1, the next step in the development and deployment process is step 103, where the software developers actually write the code....

> In step 104, the developers unit test the new code to ensure it provides the functionality defined in steps 100 and 102. **This testing is generally performed in isolation, that is, the testing goal need not include testing the new code's interaction with other software components.** The development and deployment tool is used to create a "sandbox" environment in which to complete this testing. A sandbox environment is a unit testing system allowing a developer to build as many times as he deems necessary without CM assistance....
>
> Additional build information collected by the development and deployment tool include the component to be tested and the environment in which to perform the testing....

(Emphasis added.)

Dougherty fails to teach "responsive to a determination to test the new software module in the test data processing system in combination with the existing set of software modules, identifying an operating environment of the data processing system" because *Dougherty* is silent as to this feature. In fact, *Dougherty* teaches a system and method for a web-based application development and deployment tracking tool. *Doughertys*' invention is concerned with applications still in the development stage, whereas claim 1 is concerned with the compatibility of software modules. *Dougherty* at column 7, lines 60-62 states that the "testing is generally performed in isolation, that is the testing goal need not include testing the new code's interaction with other software components." This statement teaches that the new code is tested in isolation from other components in the same software application. *Dougherty* is silent as to testing software applications against other software applications. For arguments sake, even if *Dougherty* taught that the new code is tested in isolation from different software applications *Dougherty* would still fail to teach claim 1 because the fact that testing is done in isolation teaches away from claim 1.

2. "installing the new software module, the identified environment, and the existing set of software modules on the test data processing system"

An element of claim 1 recites "installing the new software module, the identified environment, and the existing set of software modules on the test data processing system[.]"

The Examiner states that *Dougherty* at column 8, lines 9-47 and column 9, lines 30-60 teaches this element of claim 1. *Dougherty* at column 8, lines 9-47 and column 9, lines 30-60 state the following:

> Additional build information collected by the development and deployment tool include the component to be tested and the environment in which to perform the testing. This information is entered via drop-down lists 1102 and 1104. Additionally, a new source view may be entered into option box 1106. Again, the developer submits the information using "Submit" button 1108. At this point, the development and deployment tracking tool will compile the necessary code, deploy it to the correct environment and install the software in the environment....
>
> Once the developer is satisfied with the unit testing of the new application, he must submit the application for testing in an integrated environment in step 106. This testing generally done under the developer's control or the control of the developer's group. This integrated testing environment is also referred to herein as "common development" or "common dev." Building to the common development environment, is the first step in moving a release to production....

Column 8, lines 9-47.

> As described above, if the component is approved the process moves on to step 112 where the application is deployed for system testing. The purpose of the system testing is to check the application for functionality and operability in a test environment. Such testing is typically performed by a quality assurance (QA) group, separate from the development group. The application is deployed to the test system by selecting button 15 (shown in FIG. 2) and choosing option "Deploy to SysTest" from menu 1502 (shown in FIG. 15). When this option is selected, a form such as shown in FIG. 16 is presented to the user. The user then selects the component to be tested in the system test via drop-down list 1602 and submits the request via "Submit" button 1604. Drop-down list 1602 displays all components that are ready and available to be pushed to system test environment. Again, the developer submits the information using "Submit" button 1604.
>
> In step 114 (shown in FIG. 1) the application undergoes system testing to verify the application's functionality as described above. In step 116, the evaluators grade the application's performance using a view such as shown in FIG. 14. If the application fails this testing, it is sent back to the application developers in step 103. Otherwise, the application is ready for deployment to a pre-production system in step 118. In a preferred embodiment, a pre-production system replicates as much of the operational (i.e., production) system as possible. One purpose of testing the application in a pre-production environment is to optimize the performance of the application and to ensure that the application will be durable when it is moved to the operation environment.

Column 9, lines 30-60.

Dougherty fails to teach "installing the new software module, the identified environment, and the existing set of software modules on the test data processing system" because *Dougherty* is silent as to this feature. In fact, *Dougherty* teaches a system and method for a web-based application development and deployment tracking tool. *Doughertys*' invention is concerned with applications still in the development stage. For arguments sake, even if *Dougherty* was not limited to application still in the development stage, *Dougherty* is still silent as to installing a new software module, the identified environment, and the existing set of software modules on the test data processing system. *Dougherty* at column 9, lines 54-60 states that "[i]n a preferred embodiment, a pre-production system replicates as much of the operational (i.e., production) system as possible. One purpose of testing the application in a pre-production environment is to optimize the performance of the application and to ensure that the application will be durable when it is moved to the operation environment." *Dougherty* teaches that the pre-production system replicates the operating environment as much as possible. An operating environment does not include an existing set of software modules. Therefore, *Dougherty* is silent as to installing an existing set of software modules, and thus, fails to teach this element of claim 1.

3. "responsive to a test result indicating that the new software module is compatible with the existing set of software modules, adding a new combination indicating the compatibility to the knowledge base"

An element of claim 1 recites "responsive to a test result indicating that the new software module is compatible with the existing set of software modules, adding a new combination indicating the compatibility to the knowledge base[.]"

The Examiner states that *Dougherty* at column 4, lines 6-56; column 7, line 52 through column 8, line 28; and column 9, line 47 through column 10 line 10 teaches this element of claim 1. *Dougherty* at column 4, lines 6-56; column 7, line 52 through column 8, line 28; and column 9, line 47 through column 10 line 10 states the following:

> The development and deployment tool described herein provides a framework for managing and enforcing a software application development and deployment process. **The tool provides users with a system and method for automating the software builds for each phase of the development and deployment cycle. As known in the art, a software build is the result of the compiling of any necessary code....**
>
> ...

> As indicated above, the development and deployment tool may also be used to enforce the configuration management (CM) policy for an organization. For example, if the CM policy requires all code to be tested in numerous environments, the tool may be used to graduate the application to each level as it passes the earlier testing levels. If the application fails any of the required tests, the development and deployment tool can be used to send the build back to the developers for corrections.
>
> . . . Also, as a web-based tool, the system is not limited to any one hardware configuration and may be installed and run, for example, on a UNIX-based server, a Windows NT-based server, or other server system. Similarly, users may access the development and deployment tool from any web browser.
>
> FIG. 1A is a flow diagram showing the steps that may be performed in an exemplary web-based software development and deployment tool according to the present invention. In step 100, a new project is defined. This step is typically carried out by the business or other functional areas within the organization. During this step, the end-users define what they what want the new (or updated) software application to do. For example, the end-users may want to change an ordering page in a web-based store to use a new contract for interfacing with the back-end systems.

Column 4, lines 6-56 (emphasis added).

> Referring back to FIG. 1, the next step in the development and deployment process is step 103, where the software developers actually write the code....
>
> In step 104, the developers unit test the new code to ensure it provides the functionality defined in steps 100 and 102. **This testing is generally performed in isolation, that is, the testing goal need not include testing the new code's interaction with other software components.** The development and deployment tool is used to create a "sandbox" environment in which to complete this testing. A sandbox environment is a unit testing system allowing a developer to build as many times as he deems necessary without CM assistance....
>
> Additional build information collected by the development and deployment tool include the component to be tested and the environment in which to perform the testing....

Column 7, line 52 through column 8, line 28 (emphasis added.)

> In step 114 (shown in FIG. 1) the application undergoes system testing to verify the application's functionality as described above. In step 116, the evaluators grade the application's performance using a view such as shown in FIG. 14. If the

application fails this testing, it is sent back to the application developers in step 103. Otherwise, the application is ready for deployment to a pre-production system in step 118. In a preferred embodiment, a pre-production system replicates as much of the operational (i.e., production) system as possible. One purpose of testing the application in a pre-production environment is to optimize the performance of the application and to ensure that the application will be durable when it is moved to the operation environment.

The development and deployment tool is used by the configuration management group to deploy the application to the pre-production system. This is accomplished by selecting button 15 (shown in FIG. 2) and choosing option "Deploy to Pre-Production" from menu 1502 (shown in FIG. 15). When this option is selected, a form such as shown in FIG. 17 is presented to the user. This form is essentially the same as that shown in FIG. 16, except that the component selected in drop-down list 1602 will be built for the pre-production equipment.

In step 120, the new application is tested on the pre-production equipment. In step 122, the evaluators grade the application as described above. If the application passes the pre-production equipment test, the final step is to deploy the application to the production system in step 124. Otherwise, if the application fails the pre-production testing, it is sent back to the application developers in step 103.

Column 9, line 47 through column 10, line 10.

Dougherty fails to teach "responsive to a test result indicating that the new software module is compatible with the existing set of software modules, adding a new combination indicating the compatibility to the knowledge base" because *Dougherty* is silent as to this element of claim 1.

First, *Doughertys*' invention is concerned with applications still in the development stage, whereas claim 1 is concerned with the compatibility of software modules. *Dougherty* at column 4, lines 9-13 state that "[t]he tool provides users with a system and method for automating the software builds for each phase of the development and deployment cycle. As known in the art, a software build is the result of the compiling of any necessary code."

Second, even if *Dougherty* was not limited to application still in the development stage, *Dougherty* is silent as to a test result indicating that the new software module is compatible with the existing set of software modules. *Dougherty* at column 9, lines 54-60 states that "[i]n a preferred embodiment, a pre-production system replicates as much of the operational (i.e., production) system as possible. One purpose of testing the application in a pre-production environment is to optimize the performance of the application and to ensure that the application will be durable when it is moved to the operation environment." *Dougherty* teaches that the pre-production system replicates the operating environment as much as possible. However, *Dougherty* is silent as to a new software module being compatible with an existing set of software modules.

Third, *Dougherty* at column 7, lines 60-62 states that the "testing is generally performed in isolation, that is the testing goal need not include testing the new code's interaction with other software components." This statement teaches that the new code is tested in isolation from other components in the same software application. *Dougherty* is silent as to testing software applications against other software applications. For arguments sake, even if *Dougherty* taught that the new code is tested in isolation from different software applications *Dougherty* would still fail to teach claim 1 because the fact that testing is done in isolation teaches away from claim 1.

Lastly, *Dougherty* is silent as to adding a new combination indicating compatibility to the knowledge base in response to test results. Therefore, *Dougherty* fails to teach each and every element of claim 1.

B. There Must be Some Motivation or Incentive to Modify or Combine the Prior Art

For the sake of argument, even if all of the elements of the claims presented actually existed in the combination of *Sprecher* and *Dougherty*, for the rejected claims to be obvious, there must be some motivation or incentive to one of ordinary skill in the art to modify or combine the reference teachings to achieve the present invention. Often, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue. *KSR Int'l. Co. v. Teleflex, Inc.*, No. 04-1350 (U.S. Apr. 30, 2007). Rejections on obviousness grounds cannot be sustained by mere conclusory statements. Instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness. *Id.* (citing *In re Kahn*, 441 F.3d 977, 988 (CA Fed. 2006)).

The Examiner fails to provide a rational underpinning to support the legal conclusion of obviousness of the claims in view of the combination of the references when considered as a whole.

Sprecher teaches determining whether pre-requisite software is installed on a system before installing a software module. *Dougherty* teaches a system and method for a web-based application development and deployment tool, wherein the application is still in the development stage. In contrast, the present invention embodied in claim 1 is directed towards testing the compatibility of software modules, wherein a new software module's compatibility is tested against an existing set of software modules. The Examiner fails to state why one of ordinary skill in the art would look to modify *Sprecher* with *Dougherty* in order to test the compatibility of software modules.

The Examiner has not provided any motivation from the prior art or from the knowledge possessed by a person having ordinary skill in the art to make all the necessary modifications to the reference teachings to achieve the present invention. The Examiner has not provided a basis to support the legal conclusion of obviousness, but rather has simply relied on hindsight with the benefit of Applicants' disclosure to develop an incentive for the changes which, in fact, would not be obvious to one of ordinary skill in the art at the time the invention was made. The Examiner has only provided a conclusory statement and then assumed the legal conclusion without the required analysis. Therefore, the Examiner's statement does not provide a rational underpinning to support the legal conclusion of obviousness, as required by *KSR Int'l.* Accordingly, the Examiner failed to state a prima facie obviousness rejection against the canceled claims and no prima facie obviousness rejection can be stated against the presently claimed invention.

C. Remaining Claims Distinguish Over the Cited References

Each dependent claim is considered to be patentably distinguishable over the art for at least the same reasons given in support thereof.

Claims 20 and 26 are independent claims that incorporate the patentable subject matter of claim 1, and are considered to distinguish over the art for at least the reasons given in support thereof.

Claims 2-5, and 7 depend from claim 1. Claims 21-25 depend from claim 20. Claims 27-31 depend from claim 26. Each dependent claim is considered to be patentably distinguishable over the art for at least the same reasons given in support thereof.

Therefore, the rejection of claims 1-5 and 7 under 35 U.S.C. § 103 has been overcome.

II. Conclusion

It is respectfully urged that the subject application is patentable over cited references and is now in condition for allowance.

The examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: January 26, 2009

Respectfully submitted,

/Gerald H. Glanzman/

Gerald H. Glanzman
Reg. No. 25,035
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants